FuzzyLP: An R Package for Solving Fuzzy Linear Programming Problems

Pablo J. Villacorta University of Granada
Carlos A. Rabelo University of Granada
David A. Pelta University of Granada
José L. Verdegay University of Granada

Abstract

An inherent limitation of Linear Programming is the need to know precisely all the conditions concerning the problem being modeled. This is not always possible as there exist uncertainty situations which require a more suitable approach. Fuzzy Linear Programming allows working with imprecise data and constraints, leading to more realistic models. Despite being a consolidated field with more than 30 years of existence, almost no software has been developed for public use that solves fuzzy linear programming problems. Here we present an open-source R package to deal with fuzzy constraints, fuzzy costs and fuzzy coefficients in linear programming. The theoretical foundations for solving each type of problem are introduced first, followed by code examples. The package is accompanied by a user manual and can be freely downloaded, employed and extended by any R user.

Keywords: fuzzy sets, fuzzy linear programming, linear programming, R.

1. Introduction

Linear Programming (LP) is one of the main branches of Operational Research. It is composed of optimization models whose objective function and constraints are linear on the decision variables. Due to their simplicity, they have often been used for solving a wide variety of problems in sciences and engineering, enabling important benefits and savings for companies and organizations. Efficient algorithms exist for this problem, such as simplex, created in 1947 Dantzig (1987).

One limitation of LP is the requirement to know precisely all the parameters of the problem. This is sometimes not possible due to risk or uncertainty in some data, which can be coped using fuzzy numbers. Such situations are very common. Fuzzy Linear Programming (FLP) Fedrizzi, Kacprzyk, and Verdegay (1991), as a particular case of the more broad field of Fuzzy Convex Optimization Silva, Cruz, Verdegay, and Yamakami (2010), allows working with imprecision in both the coefficients and the constraints, yielding more realistic models. We can distinguish three cases:

- Problems with fuzzy constraints problems, where some degree of violation of the constraints is allowed.
- Problems with fuzzy costs, where the coefficients of the objective function are fuzzy.
- Problems with fuzzy coefficients, where the coefficients of the constraints are fuzzy.

although it is usual to find problems that combine more than one of these.

A list of applications of FLP can be found in Rommelfanger (1996); we summarize some of them below.

- Agricultural economy: water usage and water scheduling in agriculture, diet problems, optimization of farms structure, allocation of regional resources.
- Banking: assessing financial assets, portfolio problems, investment.
- Environment: regulation of air pollution, energy production models.
- Manufacturing: aggregated production scheduling, machine optimization, optimal allocation for metal manufacturing, optimal system design, raw oil processing.
- Personnel management and coordination.
- Transportation problems, track routing problems

Further examples include industrial production planning Vasant (2003), optimal water allocation with environmental constraints Tsakiris and Spiliotis (2004), optimal cattle diets Cadenas, Pelta, Pelta, and Verdegay (2004), supply chain management under uncertainty Peidro, Mula, and Poler (2007), optimization of football team resources to maximize performance Cadenas, Liern, Sala, and Verdegay (2010), and energy management Sadeghi and Hosseini (2013), just to cite a few.

Although a very large number of approaches can be found in the literature, software solutions are scarce. As highlighted in Sadeghi and Hosseini (2013), researchers usually focus on toy examples to demonstrate their novel mathematical solution methods. In Sadeghi and Hosseini (2013), a proposal on a fuzzy version of GAMS Rosenthal (2014), a popular algebraic modeling language for crisp mathematical programming, is briefly outlined but no further development was done. Apart from this, there exist (a) a non-general software aimed at solving a concrete problem, such as SACRA Cadenas, Pelta, and Verdegay (2003) for cattle diet optimization, and (b) a decision support system, called PROBO Cadenas and Verdegay (1995), written in Pascal which is difficult to integrate with current software environments.

This contribution is aimed at partially filling this gap by presenting a ready-to-use implementation in a modern language so that existing FLP methods can be used by the R community without much effort[1]. At the same time, we hope our work will contribute to further software developments in this direction to make fuzzy mathematical programming models in general readily available for researchers and practitioners from other areas, spreading this methodology beyond the fuzzy community.

The remainder of the present work is structured as follows. Section 2 provides background on some key concepts concerning fuzzy numbers, although the reader is assumed already familiar with the foundations of fuzzy sets. Section 3 describes the mathematical models implemented in our package, along with fragments of R code showing their corresponding implementation and usage. Section 4 is devoted to conclusions and further work.

[1]FuzzyLP can be downloaded from `http://decsai.ugr.es/~pjvi/FuzzyLP_0.1-3.tar.gz`

Packages used Two contributed packages have been used in implementing our `FuzzyLP` package:

- The package `FuzzyNumbers` Gagolewski (2014) provides classes and methods for working with fuzzy numbers. More precisely, it allows representing generic fuzzy numbers as well as some particular types (triangular, trapezoidal, piecewise linear). It offers several defuzzification functions as well as functions to approximate general fuzzy numbers using piecewise linear fuzzy numbers. Basic arithmetic can also be done with fuzzy numbers. Finally, it also has plotting facilities. According to the author, future versions may include random fuzzy number generation, aggregation and sorting.
- Package `ROI` (R Optimization Infrastructure Theussl, Meyer, and Hornik (2010)). Since FLP ultimately relies on crisp LP solving algorithms, this package has been used to accomplish such task. A large number of optimization packages are available in R; see the R Task View on Optimization[2]. Among them, `ROI` is an attempt to provide a unified interface for any optimization problem. For this reason, along with the possibility that future versions of our `FuzzyLP` package need to deal with other non-linear fuzzy optimization problems which eventually rely on more complex crisp solvers that ROI already offers, we have chosen this package.

2. Fuzzy numbers

We assume the reader is familiar with the basics of fuzzy sets and fuzzy numbers, and therefore we only review those concepts that are highly relevant for our later exposition. The interested reader may refer to Dubois and Prade (1980); Negoita and Ralescu (1975) for an introduction to the topic.

Definition 1 *A fuzzy number Dubois and Prade (1978) $\tilde{A}$ is a convex and normalized subset of the real line such that*

i) $\forall \alpha \in [0,1], \tilde{A}_\alpha = \{x \in \mathbb{R} | \mu_{\tilde{A}} \geq \alpha\}$ *(the α-cuts of $\tilde{A}$) is a convex set.*

ii) $\mu_{\tilde{A}}$ *is upper semi-continuous*

iii) $Supp(\tilde{A}) = \{x \in \mathbb{R} | \mu_{\tilde{A}} > 0\}$ *is a bounded set on $\mathbb{R}$.*

Definition 2 *Given $r < u \leq U < R \in \mathbb{R}$, a Trapezoidal Fuzzy Number (TrFN) $\tilde{u} = (r, u, U, R)$ is defined as:*

$$\mu_{\tilde{u}}(x) = \begin{cases} \frac{x-r}{u-r} & \text{if} \quad r \leq x \leq u \\ 1 & \text{if} \quad u \leq x \leq U \\ \frac{R-x}{R-U} & \text{if} \quad U \leq x \leq R \\ 0 & \text{otherwise} \end{cases}$$

[2] http://cran.r-project.org/web/views/Optimization.html

Proposition 1 (Linear combination of TrFNs Tanaka, Ichihashi, and Asai (1984)) *Let* $\tilde{B} = \sum_{j=1}^{n} \tilde{u}_j \cdot a_j$ *where* $a_j \in \mathbb{R}$, $a_j \geq 0$, $j = 1, \ldots, n$, *and let* $\tilde{u}_j$, $j = 1, \ldots, n$ *be TrFNs defined by* $\tilde{u}_j = (r_j, u_j, U_j, R_j)$. *Then the membership function of* $\tilde{B}$ *is*

$$\mu_{\tilde{B}}(x) = \begin{cases} \frac{x - \mathbf{ra}}{\mathbf{ua} - \mathbf{ra}} & \text{if} \quad \mathbf{ra} \leq x \leq \mathbf{ua} \\ 1 & \text{if} \quad \mathbf{ua} \leq x \leq \mathbf{Ua} \\ \frac{\mathbf{Ra} - x}{\mathbf{Ra} - \mathbf{Ua}} & \text{if} \quad \mathbf{Ua} \leq x \leq \mathbf{Ra} \\ 0 & \text{otherwise} \end{cases}$$

where $\mathbf{r} = (r_1, \ldots, r_n)$, $\mathbf{u} = (u_1, \ldots, u_n)$, $\mathbf{U} = (U_1, \ldots, U_n)$ *and* $\mathbf{R} = (R_1, \ldots, R_n)$. *In other words,* $\tilde{B} = \sum_{j=1}^{n} \tilde{u}_j \cdot a_j = (\mathbf{ra}, \mathbf{ua}, \mathbf{Ua}, \mathbf{Ra})$.

2.1. Operations with fuzzy numbers

In the remainder of this work, only TrFNs will be used exclusively. For this reason and due to space constraints, we only review operations with TrFNs. It is well known that the product and quotient operations with TrFNs do not yield another TrFN. However, only addition, subtraction and product by a scalar are needed for the FLP algorithms implemented here. Hence we focus on those operations.

Proposition 2 *Let* $\tilde{x}_i = (r_i, u_i, U_i, R_i)$, $i = 1, 2$, *two TrFNs. Then*

i) $\tilde{x}_1 + \tilde{x}_2 = (r_1 + r_2, u_1 + u_2, U_1 + U_2, R_1 + R_2)$

ii) $\tilde{x}_1 - \tilde{x}_2 = (r_1 - R_2, u_1 - U_2, U_1 - u_2, R_1 - r_2)$

iii) If $a \in \mathbb{R}^+$, $a \cdot \tilde{x}_1 = a \cdot (r_1, u_1, U_1, R_1) = (a \cdot r_1, a \cdot u_1, a \cdot U_1, a \cdot R_1)$

iv) If $a \in \mathbb{R}^-$, $a \cdot \tilde{x}_1 = a \cdot (r_1, u_1, U_1, R_1) = (a \cdot R_1, a \cdot U_1, a \cdot u_1, a \cdot r_1)$

2.2. Comparison of fuzzy numbers

As will be explained later, FLP ultimately requires comparing fuzzy numbers. This is a broad topic by itself, and a lot of proposals have been published; see for instance Dubois and Prade (1983); Prade, Yager, and Dubois (1993). Here the method based on ordering functions has been applied, although the code is prepared for other customized comparison functions implemented by the user that can be passed as an argument.

Ordering functions Let $\mathcal{F}(\mathbb{R})$ the set of fuzzy numbers on $\mathbb{R}$. An ordering (or defuzzification) function is an application $g : \mathcal{F}(\mathbb{R}) \to \mathbb{R}$ so that fuzzy numbers are sorted according to their corresponding defuzzified real numbers. Therefore, given $\tilde{A}, \tilde{B} \in \mathcal{F}(\mathbb{R})$, we consider $\tilde{A} < \tilde{B} \Leftrightarrow g(\tilde{A}) < g(\tilde{B})$; $\tilde{A} > \tilde{B} \Leftrightarrow g(\tilde{A}) > g(\tilde{B})$; and $\tilde{A} = \tilde{B} \Leftrightarrow g(\tilde{A}) = g(\tilde{B})$. Function g is called a linear ordering function if (a) $\forall\ \tilde{A}, \tilde{B} \in \mathcal{F}(\mathbb{R})$, $g(\tilde{A} + \tilde{B}) = g(\tilde{A}) + g(\tilde{B})$, and (b) $\forall\ r \in \mathbb{R}$, $r > 0$ and $\forall\ \tilde{A} \in \mathcal{F}(\mathbb{R})$, $g(r \cdot \tilde{A}) = r \cdot g(\tilde{A})$.

The following linear ordering functions have been implemented:

1. First Yager's index Yager (1978):

$$g(\tilde{u}) = \frac{\int_S h(z)\mu_{\tilde{u}}(z)dz}{\int_S \mu_{\tilde{u}}(z)dz} \tag{1}$$

where $S = \text{supp}(\tilde{u})$ and $h(z)$ is a measure of importance of each value z. Taking $h(z) = z$ and assuming TrFNs, this simplifies to

$$g(\tilde{u}) = \frac{1}{3}\frac{(U^2 - u^2) + (R^2 - r^2) + (RU - ru)}{(U - u) + (R - r)} \tag{2}$$

2. Third Yager's index Yager (1978):

$$g(\tilde{u}) = \int_0^1 M(\tilde{u}_\alpha)d\alpha \tag{3}$$

where $\tilde{u}_\alpha$ is an α-cut of $\tilde{u}$ and $M(\tilde{u}_\alpha)$ the mean value of the elements in $\tilde{u}_\alpha$. When using TrFNs the above expression simplifies to

$$g(\tilde{u}) = \frac{U + u + R + r}{4} \tag{4}$$

3. Adamo relation Adamo (1980). For a fixed $\alpha \in [0, 1]$:

$$g_\alpha(\tilde{u}) = \max\{x \ / \ \mu_{\tilde{u}}(x) \geq \alpha\} \tag{5}$$

which for TrFNs simplifies to

$$g_\alpha(\tilde{u}) = R - \alpha(R - U) \tag{6}$$

4. Average relation González (1990). For TrFNs:

$$g_t^\lambda(\tilde{u}) = u - \frac{u - r}{t + 1} + \lambda\left(U - u + \frac{(R - r) - (U - u)}{t + 1}\right), \ \lambda \in [0, 1], \ t \geq 0 \tag{7}$$

where λ represents the degree of optimism to be selected by the decision-maker (larger λ corresponds to a more optimist decision-maker).

3. Fuzzy Linear Programming

A crisp LP problem consists in maximizing/minimizing a function subject to constraints over the variables:

$$\begin{aligned} \max z &= \mathbf{cx} \\ s.t. : \mathbf{Ax} &\leq \mathbf{b} \\ \mathbf{x} &\geq \mathbf{0} \end{aligned}$$

where $\mathbf{A} \in \mathcal{M}_{mxn}(\mathbb{R})$ is a matrix of real numbers, $\mathbf{c} \in \mathbb{R}^n$ is a cost vector and $\mathbf{b} \in \mathbb{R}^m$ a vector.

In the above formulation, all coefficients are assumed to be perfectly known. However, this is not the case in many real cases of application. There may be uncertainty concerning some coefficients, or they may come from an (approximate) estimation by a human expert/decision maker who will possibly be more confident when expressing her knowledge in linguistic terms of the natural language Zadeh (1975). Very often, when a person is asked to express her expertise in strictly numerical values, he/she feels like being forced to commit an error, hence the use of natural language, supported by fuzzy numbers to do computations, might be more suitable. Optimization problems with fuzzy quantities were first presented in Bellman and Zadeh (1970). Key concepts such as fuzzy constraint and fuzzy goal, which we will explain in detail later in this section, were conceived there.

In the next sections we explain in detail the three FLP models and several solution methods implemented in our package.

3.1. Fuzzy constraints

We consider the case where the decision maker can accept a violation of the constraints up to a certain degree he/she establishes. This can be formalized for each constraint as

$$a_i x \leq_f b_i,\ i = 1, ..., m$$

and can be modeled using a membership function

$$\mu_i : \mathbb{R} \to [0,1], \quad \mu_i(x) = \begin{cases} 1 & \text{if} \quad x \leq b_i \\ f_i(x) & \text{if} \quad b_i \leq x \leq b_i + t_i \\ 0 & \text{if} \quad x \geq b_i + t_i \end{cases} \tag{8}$$

where the f_i are continuous, non-increasing functions. Membership functions μ_i capture the fact that the decision maker tolerates a certain degree of violation of each constraint, up to a value of $b_i + t_i$. For each $x \in \mathbb{R}$, $\mu_i(x)$ stands for the degree of fulfillment of the i-th constraint for that x. The problem to be solved is

$$\begin{aligned} \max z &= \mathbf{cx} \\ s.t. : \mathbf{Ax} &\leq_f \mathbf{b} \\ \mathbf{x} &\geq \mathbf{0} \end{aligned}$$

To illustrate the use of the functions implemented to solve this type of problems, we will use the fuzzy constraints problem shown on the left, which can be transformed into the problem on the right if the membership functions are assumed linear with maximum tolerances of 5 and 6.

$$\begin{aligned} \max z = 3x_1 + x_2 & & \max z = 3x_1 + x_2 \\ s.t. : 1.875x_1 - 1.5x_2 \leq_f 4 \quad & \Rightarrow \quad s.t. : 1.875x_1 - 1.5x_2 \leq 4 + 5(1-\alpha) \\ 4.75x_1 + 2.125x_2 \leq_f 14.5 \quad & \Rightarrow \quad 4.75x_1 + 2.125x_2 \leq 14.5 + 6(1-\alpha) \\ x_i \geq 0, i = 1,2,3 & & \alpha_i \geq 0, i = 1,2,3 \end{aligned}$$

The following R commands create the necessary objects:

```
> objective<-c(3, 1)
> A<-matrix(c(1.875, -1.5, 4.75, 2.125), nrow = 2, byrow = T)
> dir = c("<=", "<=") # direction of the inequalities
> b<-c(4, 14.5)
> t<-c(5, 6) # tolerances
```

Different solutions have been proposed. The solution given in Verdegay (1982) generalizes those given before by Tanaka, Okuda, and Asai (1974) and Zimmermann (1976), which are obtained as particular cases depending of the value of a parameter. We implement four solution methods, whose names begin with `FCLP` for Fuzzy Constraints Linear Program

Method 1: Verdegay's approach Verdegay (1982) proved, via the representation theorem, that the problem can be solved by solving the following Parametric Linear Programming problem:

$$\begin{aligned} \max z &= \mathbf{cx} \\ s.t. : \mathbf{Ax} &\leq \mathbf{g}(\alpha) \\ \mathbf{x} &\geq \mathbf{0}, \alpha \in [0,1] \end{aligned}$$

where $\mathbf{g}(\alpha) = (g_1(\alpha), \ldots, g_m(\alpha)) \in \mathbb{R}^m$, with $g_i = f_i^{-1}$.
If the f_i are linear, the problem simplifies to

$$\begin{aligned} \max z &= \mathbf{cx} \\ s.t. : \mathbf{Ax} &\leq \mathbf{b} + \mathbf{t}(1-\alpha) \\ \mathbf{x} &\geq \mathbf{0}, \alpha \in [0,1] \end{aligned}$$

with $\mathbf{t} = (t_1, \ldots, t_m) \in \mathbb{R}^m$.
It has been proved Delgado, Herrera, Verdegay, and Vila (1993) that by solving a model with linear f_i, it is possible to obtain the solution to the same fuzzy constraints problem as if it had been modeled with non-linear functions, hence no generality is loss when assuming linear functions for the fuzzy constraints.
Two R functions implement this approach in our package:

- `FCLP.fixedBeta` which, fixed a value for α (called β in our function), solves the model. In the example below, $\beta = 0.5$.

```
> FCLP.fixedBeta(objective, A, dir, b, t, beta=0.5, T, T)
```

```
[1] "Solution is optimal."
     beta       x1        x2 objective
[1,]  0.5 3.606188 0.1744023  10.99297
```

- `FCLP.sampledBeta` samples α in the interval $[0,1]$ and solves the model for every sampled value. In the example below we set a step size of 0.25 for sampling α, which yields five α-cuts, for $\alpha = \{0, 0.25, 0.5, 0.75, 1\}$:

```
> FCLP.sampledBeta(objective, A, dir, b, t, T, min=0,
+                  max=1, step=0.25)
     beta       x1        x2 objective
[1,] 0.00 4.315789 0.0000000 12.947368
[2,] 0.25 4.000000 0.0000000 12.000000
[3,] 0.50 3.606188 0.1744023 10.992968
[4,] 0.75 3.164557 0.4556962  9.949367
[5,] 1.00 2.722925 0.7369902  8.905767
```

Method 2: Zimmermann's approach In Zimmermann (1976, 1978) the author discusses the case in which the decision maker is satisfied with a solution that achieves a goal $z_0 \in \mathbb{R}$ for the objective function that, despite not being optimal, minimizes the degree of violation of the constraints. The original formulation is shown on the left. This problem is equivalent to the one on the right, where an additional fuzzy linear constraint has been added. Please notice the new constraint can be thought as embedded in the general expression of the linear constraints, $\mathbf{Ax} \leq_{\mathbf{f}} \mathbf{b}$ if we assume that $\mathbf{A}$ and $\mathbf{b}$ include an extra row for $-\mathbf{c}$ and an element $-z_0$ respectively.

$$\begin{array}{rcl} \max \mathbf{cx} \geq_f z_0 & & \max z = \mathbf{cx} \\ s.t. : \mathbf{Ax} \leq_f \mathbf{b} & \Rightarrow & s.t. : \mathbf{cx} \geq_f z_0 \\ \mathbf{x} \geq \mathbf{0} & & \mathbf{Ax} \leq_f \mathbf{b}, \mathbf{x} \geq \mathbf{0} \end{array}$$

When the membership functions of the constraints and the objective are linear the above problem simplifies to

$$\begin{array}{rcl} \max & \alpha & \\ s.t. : \mathbf{cx} & \geq & z_0 - t_0(1-\alpha) \\ \mathbf{Ax} & \leq & \mathbf{b} + \mathbf{t}(1-\alpha) \\ \mathbf{x} & \geq & \mathbf{0}, \alpha \in [0,1] \end{array}$$

Two R functions implement this approach:

- `FCLP.classicObjective` for the case that the goal z_0 is crisp.

Goal attainable ($z_0 = 11$):

```
> FCLP.classicObjective(objective, A, dir, b, t, z0=11, TRUE)

Bound reached, FCLP.fixedBeta with beta = 0.4983154 may
obtain better results.
          beta       x1        x2 objective
[1,] 0.4983154 3.609164 0.1725067        11
```

Goal not attainable ($z_0 = 14$):

```
> FCLP.classicObjective(objective, A, dir, b, t, z0=14, TRUE)

[1] "Minimal bound not reached."
NULL
```

- `FCLP.fuzzyObjective` for a goal ($z_0 = 14$) on which we admit certain tolerance ($t_0 = 2$).

```
> FCLP.fuzzyObjective(objective, A, dir, b, t, z0=14, t0=2, T)

Bound reached, FCLP.fixedBeta with beta = 0.1636364 may obtain
better results.
          beta       x1 x2 objective
[1,] 0.1636364 4.109091  0  12.32727
```

Actually `FCLP.fuzzyObjective` generalizes `FCLP.classicObjective`. The latter simply calls the former setting the tolerance $t_0 = 0$.

Method 3: Werners's approach Following Zimmermann's proposal, it may occur that the decision maker does not want to provide the goal or the tolerance, or does not have an estimate. Werners Werners (1987) proposes two extreme points:

$$Z^0 = \inf\{\max_{\mathbf{x}\in X} \mathbf{cx}\}, \qquad Z^1 = \sup\{\max_{\mathbf{x}\in X} \mathbf{cx}\} \tag{9}$$

with $X = \{\mathbf{x} \in \mathbb{R}^n \ / \ \mathbf{Ax} \leq_f \mathbf{b},\ \mathbf{x} \geq \mathbf{0}\}$. Taking $z_0 = Z^0$ and $t_0 = Z^1 - Z^0$ and assuming linear functions for the constraints and the objective, the new problem to be solved can be formulated as

$$\begin{array}{rcl} \max & \alpha & \\ s.t.: \mathbf{cx} & \geq & Z^0 - (Z^1 - Z^0)(1-\alpha) \\ \mathbf{Ax} & \leq & \mathbf{b} + \mathbf{t}(1-\alpha) \\ \mathbf{x} & \geq & \mathbf{0}, \quad \alpha \in [0,1] \end{array}$$

which is a particularization of Zimmermann's formulation with the aforementioned z_0 and t_0.

This method is implemented by function `FCLP.fuzzyUndefinedObjective`, in which the goal and tolerance are estimated and then, `FCLP.fuzzyObjective` is called:

```
> FCLP.fuzzyUndefinedObjective(objective, A, dir, b, t, TRUE)

[1] "Using bound = "   "12.9473684210526"
[1] "Using tolerance = " "4.04160189503294"
          beta       x1        x2 objective
[1,] 0.5080818 3.591912 0.1834957  10.95923
```

Method 4: Tanaka's approach Tanaka Tanaka *et al.* (1974) proposed normalizing the objective function $z = \mathbf{cx}$. Let M be the optimum when the problem is solved considering crisp constraints. Then

$$f : \mathbb{R}^n \to [0,1], \quad f(x) = \begin{cases} 1 & \text{if} \quad \mathbf{cx} > M \\ \frac{\mathbf{cx}}{M} & \text{if} \quad \mathbf{cx} \leq M \end{cases}$$

The new problem to be solved is

$$\begin{array}{rcl} \max & \alpha & \\ s.t. : \dfrac{\mathbf{cx}}{M} & \geq & \alpha \\ \mathbf{Ax} & \leq & \mathbf{b} + \mathbf{t}(1-\alpha) \\ \mathbf{x} & \geq & \mathbf{0},\ \alpha \in [0,1] \end{array}$$

For implementation purposes, we will do the following modification:

$$\frac{\mathbf{cx}}{M} \geq \alpha \Leftrightarrow \mathbf{cx} \geq M\alpha = M - M(1-\alpha)$$

Therefore we replace the first constraint above by $\mathbf{cx} \geq M - M(1-\alpha)$.

This approach is implemented in function `FCLP.fuzzyUndefinedNormObjective`. It first computes M and then calls `FCLP.fuzzyObjective` with $z_0 = t_0 = M$.

```
> FCLP.fuzzyUndefinedNormObjective(objective, A, dir, b, t, TRUE)

[1] "Using bound = "   "12.9473684210526"
[1] "Using tolerance = " "12.9473684210526"
          beta       x1        x2 objective
[1,] 0.7639495 3.139915 0.4713919  9.891136
```

3.2. Fuzzy costs

FLP with fuzzy costs pose uncertainty in the coefficients of the objective function, modeled as fuzzy numbers. Such problems can be stated as:

$$\begin{array}{rcl} \max z & = & \tilde{\mathbf{c}}\mathbf{x} \\ s.t. : \mathbf{Ax} \leq \mathbf{b} & & \\ \mathbf{x} \geq \mathbf{0} & & \end{array} \tag{10}$$

where $\mathbf{A} \in \mathcal{M}_{mxn}(\mathbb{R})$ is a real matrix, $\tilde{\mathbf{c}}$ is an n-dimensional vector of fuzzy numbers, and $\mathbf{b} \in \mathbb{R}^m$ is a real vector.

In Cadenas and Verdegay (1999) the costs membership functions are assumed to have the form:

$$\mu_j : \mathbb{R} \to [0,1], \quad \mu_j(x) = \begin{cases} 0 & \text{if} \quad x \leq r_j \text{ or } x \geq R_j \\ h_j(x) & \text{if} \quad r_j \leq x \leq \underline{c}_j \\ g_j(x) & \text{if} \quad \overline{c}_j \leq x \leq R_j \\ 1 & \text{if} \quad \underline{c}_j \leq x \leq \overline{c}_j \end{cases}$$

with h_j and g_j continuous, h_j strictly increasing, g_j strictly decreasing, and such that functions μ_j are continuous.

In order to demonstrate the functions of our package dealing with fuzzy costs, we will use the following example with TrFNs ((Cadenas and Verdegay 1999, pages 94,125)):

$$\begin{aligned} \max z &= (0,2,2,3)x_1 + (1,3,4,5)x_2 \\ s.t.: x_1 + 3x_2 &\leq 6 \\ x_1 + x_2 &\leq 4 \\ x_i &\geq 0, i = 1,2 \end{aligned}$$

The following R commands create the necessary objects:

```
> objective<-c(TrapezoidalFuzzyNumber(0,2,2,3), # fuzzy costs
+              TrapezoidalFuzzyNumber(1,3,4,5)) # vector
> A<-matrix(c(1, 3, 1, 1), nrow = 2, byrow=T)
> dir = c("<=", "<=") # direction of the inequalities
> b<-c(6, 4)
```

Four approaches will be presented: three of them are based on the Representation Theorem and the last one, on the comparison of fuzzy numbers.

Method 1: multi-objective approach The results of Verdegay (1982) and Delgado, Verdegay, and Vila (1987) together with the definition of TrFNs lead us to transform the above problem in the following parametric linear programming problem. Let $X = \{\mathbf{x} \in \mathbb{R}^n \ / \ \mathbf{Ax} \leq \mathbf{b},\ \mathbf{x} \geq \mathbf{0}\}$. Then

$$\begin{aligned} \max z &= \mathbf{cx} \\ s.t.: \mathbf{x} &\in X \\ \mathbf{\Phi}(1-\alpha) \leq \ \mathbf{c} &\leq \mathbf{\Psi}(1-\alpha) \\ \alpha &\in [0,1] \end{aligned} \tag{11}$$

where $\mathbf{\Phi} = (h_1^{-1}, \ldots, h_n^{-1})$ and $\mathbf{\Psi} = (g_1^{-1}, \ldots, g_n^{-1})$. Symbols h_j and g_j take part in the definition of the membership functions μ_j of every fuzzy cost $\tilde{c}_j$.

If we fix α and solve the above problem with it, the solution set is the α-cut of the solution fuzzy number, which would be completely defined by all its α-cuts as stated by the Representation Theorem. If we define $\Gamma(1-\alpha) = \{\mathbf{c} \in \mathbb{R}^n \ / \ c_i \in [\Phi_i(1-\alpha), \Psi_i(1-\alpha)]\}$, the new problem constitutes a multi-objective linear programming problem with one objective for each $\mathbf{c} \in \Gamma(1-\alpha)$. According to Bitran (1985) the problem is equivalent to the following:

$$\begin{aligned} &\max && \{\mathbf{c}^1\mathbf{x}, \ldots, \mathbf{c}^{2^n}\mathbf{x}\} \\ s.t.: \mathbf{Ax} &\leq && \mathbf{b} \\ \mathbf{x} &\geq && \mathbf{0} \\ \mathbf{c}^k &\in && E(1-\alpha),\ k = 1,2,\ldots,2^n \\ \alpha &\in && [0,1] \end{aligned}$$

where $E(1-\alpha) \subseteq \Gamma(1-\alpha)$ is the subset of those vectors whose components are the upper bounds of c_j, i.e., the Cartesian product:

$$E(1-\alpha) = \prod_{i=1}^{n} \{\Phi_i(1-\alpha), \Psi_i(1-\alpha)\}$$

This problem can be solved by any multi-objective linear programming technique. In our code, the objectives have been aggregated using a weighting vector with the same weight for every objective. The objective function thus simplifies to $\max \mathbf{c}^1\mathbf{x} + \ldots + \mathbf{c}^{2^n}\mathbf{x}$ subject to the constraints stated above.

This approach is implemented by function `FOLP.multiObj`. Since the problem has to be solved for every $\alpha \in [0,1]$, the function samples α in $[0,1]$ according to a user-specified step, and solves for each α.

```
> sal<-FOLP.multiObj(objective, A, dir, b, maximum=TRUE, min=0,
+                    max=1, step=0.25)
> sal

     alpha x1 x2 objective
[1,] 0     3  1  ?
[2,] 0.25  3  1  ?
[3,] 0.5   3  1  ?
[4,] 0.75  3  1  ?
[5,] 1     3  1  ?

> sal[,"objective"] # Display the objective column properly

[[1]]
Trapezoidal fuzzy number with:
   support=[1,14],
      core=[9,10].
 ... # output omitted for elements 2, 3, 4 and 5
```

Method 2: interval arithmetic approach Expression 11 can be viewed as a linear programming problem in which every coefficient of the objective function takes values in an interval. Therefore, the problem can be solved resorting to interval arithmetic and relations $\leq_l$, $\leq_c$, $\leq_{lc}$ introduced in Moore (1979) and Alefeld and Herzberger (1984).

Definition 3 *Let* $A = [a^l, a^u] =< a^c, a^w >$ *and* $B = [b^l, b^u] =< b^c, b^w >$ *be two intervals, where the* $<\cdot,\cdot>$ *are based on the center* c *and width* w*.*

- $A \leq_l B$ *if* $a^l \leq b^l$ *and* $a^u \leq b^u$
- $A \leq_c B$ *if* $a^c \leq b^c$ *and* $a^w \geq b^w$.
- $A \leq_{lc} B$ *if* $a^l \leq b^l$ *and* $a^c \leq b^c$
- $A <_l B$ *if* $A \leq_l B$ *and* $A \neq B$
- $A <_c B$ *if* $A \leq_c B$ *and* $A \neq B$
- $A <_{lc} B$ *if* $A \leq_{lc} B$ *and* $A \neq B$

For every $\mathbf{x} \in X, \alpha \in [0,1]$, define the intervals

$$I_j(\alpha) = [\Phi_j(1-\alpha), \Psi_j(1-\alpha)] \qquad \text{and} \qquad z(\mathbf{x},\alpha) = \sum_{j=1}^{n} x_j I_j(\alpha)$$

For each α, a solution $\mathbf{x}^*$ to the problem is one whose associated interval $z(\mathbf{x}^*,\alpha)$ is non-dominated, i.e. $\mathbf{x}^* \in X$ such that $\nexists \mathbf{x}' \in X : z(\mathbf{x}^*,\alpha) \leq_{lc} z(\mathbf{x}',\alpha)$. Since $\mathbf{x}^*$ does not have to be unique, we can define the set

$$S(1-\alpha) = \{\mathbf{x} \in X \ / \ \nexists \mathbf{x}' \in X : z(\mathbf{x},\alpha) \leq_{ic} z(\mathbf{x}',\alpha)\}$$

These sets are the α-cuts of the fuzzy solution which, according to the Representation Theorem, would yield the solution fuzzy number $\tilde{S} = \bigcup_\alpha \alpha S(1-\alpha)$.

For a fixed α, the problem of finding solutions whose associated intervals are non-dominated can be formulated as the following bi-objective problem.

$$\max\{z(\alpha) = (z^i(\mathbf{x},\alpha), z^c(\mathbf{x},\alpha)) : \mathbf{x} \in X\}$$

According to Kornbluth and Steuer (1981) this problem can be solved using weights. Let $\beta_1, \beta_2 \in [0,1] : \beta_1 + \beta_2 = 1$. The problem can be reformulated as:

$$\max\{z(\alpha) = \beta_1 z^i(\mathbf{x},\alpha) + \beta_2 z^c(\mathbf{x},\alpha) : \mathbf{x} \in X\}$$

This approach is implemented by function `FOLP.interv` which receives the problem data and weight β_1 (note β_2 can be automatically computed as 1 - β_1). The function performs samples α in $[0,1]$ with the user-specified step size. A private function computes $z(\alpha)$ from the fuzzy coefficients.

```
> sal<-FOLP.interv(objective, A, dir, b, maximum=TRUE, w1=0.7,
+                  min=0, max=1, step=0.25)
> sal
```

The structure of this variable is the same as in the previous section.

Method 3: stratified piecewise reduction In Rommelfanger, Hanuscheck, and Wolf (1989) the fuzzy cost problem is approached by modeling the uncertainty of the coefficients using embedded intervals, each with an associated possibility degree:

$$\tilde{c}_j = \{[r_j^{(k)}, R_j^{(k)}] \ / \ \alpha^{(k)}; k = 1, \ldots, p\}$$

For a given $\alpha \in [0,1]$, consider the intervals obtained from the α-cuts of the fuzzy coefficients. With a slight abuse of notation (and omitting the α that has been fixed), we will write $\tilde{c}_j = [r_j, R_j]$.

Let $\mathbf{r} = (r_1, \ldots, r_n)$, $\mathbf{R} = (R_1, \ldots, R_n)$, and consider the LP problems

$$\begin{aligned} \max z &= \mathbf{rx} \\ s.t. &: \mathbf{Ax} \leq \mathbf{b} \\ &\mathbf{x} \geq \mathbf{0} \end{aligned} \qquad\qquad \begin{aligned} \max z &= \mathbf{Rx} \\ s.t. &: \mathbf{Ax} \leq \mathbf{b} \\ &\mathbf{x} \geq \mathbf{0} \end{aligned}$$

Let $\mathbf{x}_\mathbf{r}^*$ and $\mathbf{x}_\mathbf{R}^*$ be their respective solutions.

Let $z_\mathbf{r}^* = \mathbf{r}\mathbf{x}_r^*$ and $z_\mathbf{R}^* = \mathbf{R}\mathbf{x}_\mathbf{R}^*$ be the optimal solutions of the objective functions, and let $z_\mathbf{r}^{'} = \mathbf{r}\mathbf{x}_\mathbf{R}^*$ and $z_\mathbf{R}^{'} = \mathbf{R}\mathbf{x}_\mathbf{r}^*$. Clearly $z_\mathbf{r}^* \geq z_\mathbf{r}^{'}$ and $z_\mathbf{R}^* \geq z_\mathbf{R}^{'}$.

The problem can be solved using the auxiliary problem

$$\begin{aligned} & & \max & \quad \lambda \\ & s.t.: \frac{\mathbf{r}\mathbf{x} - z_\mathbf{r}^{'}}{z_\mathbf{r}^* - z_\mathbf{r}^{'}} & \geq & \quad \lambda \\ & \frac{\mathbf{R}\mathbf{x} - z_\mathbf{R}^{'}}{z_\mathbf{R}^* - z_\mathbf{R}^{'}} & \geq & \quad \lambda \\ & \mathbf{A}\mathbf{x} \leq \mathbf{b}, \mathbf{x} \geq \mathbf{0}, \lambda & \geq & \quad 0 \end{aligned}$$

After solving the above problems for different values of α, the solution to the original fuzzy costs problem can be found as the intersection of the solutions of the auxiliary problems.

Function `FOLP.strat` computes the values $z_\mathbf{r}^*$, $z_\mathbf{r}^{'}$, $z_\mathbf{R}^*$ and $z_\mathbf{R}^{'}$ by solving the corresponding LP problems, and then solves the original problem. This has to be done separately for each value of α, therefore `FOLP.strat` samples $\alpha \in [0, 1]$ with a user-specified step size.

```
> sal <- FOLP.strat(objective, A, dir, b, maximum=TRUE, min=0,
+                   max=0.4, step=0.05)
> sal

      alpha x1  x2  lambda objective
 [1,] 0     1.5 1.5 0.5    ?
 [2,] 0.05  1.5 1.5 0.5    ?
 [3,] 0.1   1.5 1.5 0.5    ?
 [4,] 0.15  1.5 1.5 0.5    ?
 [5,] 0.2   1.5 1.5 0.5    ?
 [6,] 0.25  3   1   1      ?
 [7,] 0.3   NA  NA  NA     NA
 [8,] 0.35  NA  NA  NA     NA
 [9,] 0.4   NA  NA  NA     NA

> sal[,"objective"] # display the objective column properly

[[1]]
Trapezoidal fuzzy number with:
   support=[1.5,12],
      core=[7.5,9].
  ... # output omitted for list elements 2 to 9
```

Method 4: ordering functions The problem 10 can be transformed into a crisp one by using a linear ordering function $g : \mathcal{F}(\mathbb{R}) \to \mathbb{R}$, so that the fuzzy objective function is replaced by $\max z = g(\tilde{c}_1)x_1 + ... + g(\tilde{c}_n)x_n$, subject to the same crisp constraints.

Function `FOLP.ordFun` implements this approach. It receives the problem data and a string indicating the ordering function to be used (argument `ordf`). This can be one of the four

built-in functions, namely `"Yager1"` for the first Yager's index (Equation 2), `"Yager3"` for the third (Equation 4), `"Adamo"` for the Adamo relation (Equation 6), and `"Average"` for the average index (Equation 7). Some of them require additional arguments, as described in detail in the package documentation. If a user-defined custom linear function is to be used, the string should be `"Custom"`. The custom function is passed to `FOLP.ordFun` in the `FUN` argument. The custom function must accept at least one argument of class `FuzzyNumber`, and may also accept additional arguments that must be named when passing them to `FOLP.ordFun`. It is the user's responsibility to give them names that do not interfere with existing ones, and to care that the function is linear.

Example call using first Yager's index:

```
> sal<-FOLP.ordFun(objective, A, dir, b, maximum=TRUE,
+                  ordf="Yager1")
```

For Adamo's index, which requires an additional parameter α:

```
> sal<-FOLP.ordFun(objective, A, dir, b, maximum=TRUE,
+                  ordf="Adamo", alpha=0.5)
```

For a custom function that computes the mean of the core multiplied by another real number:

```
> custom.f <- function(tfn,a){ a * mean(core(tfn)) }
> sal<-FOLP.ordFun(objective, A, dir, b, TRUE, "Custom",
+                  FUN=custom.f, a=2)
```

3.3. General model

The most general setting is that with fuzzy costs, fuzzy coefficients in the technology matrix, and fuzzy constraints that can be violated up to a certain degree. Calling m to the number of constraints, it can be formalized as the problem on the left. This formulation can be transformed into the problem on the right, according to the Representation Theorem and assuming that the decision maker agrees with considering the same degree of satisfaction both in the fuzzy costs and in the fuzzy technological matrix[3].

$$\begin{array}{c} \max z = \tilde{\mathbf{c}}\mathbf{x} \\ s.t. : \tilde{\mathbf{a}}_i\mathbf{x} \leq_f \tilde{\mathbf{b}}_i, \\ \mathbf{x} \geq \mathbf{0} \end{array} \quad \Rightarrow \quad \begin{array}{c} \max z = \tilde{\mathbf{c}}\mathbf{x} \\ s.t. : \tilde{\mathbf{a}}_i\mathbf{x} \leq \tilde{\mathbf{b}}_i + \tilde{\mathbf{t}}_i(1-\alpha), \quad i = 1, ..., m \\ \mathbf{x} \geq \mathbf{0} \end{array}$$

where $\tilde{\mathbf{a}}_i$ and $\mathbf{b}_i (i = 1, ..., m)$ are n-dimensional vectors of fuzzy numbers, $\tilde{\mathbf{c}}$ is another n-dimensional vector of fuzzy numbers, and $\tilde{\mathbf{t}}_i$ is the fuzzy tolerance admitted for violating the i-th constraint.

Let g_1 and g_2 be two linear ordering functions for the objective and for the constraints, respectively. With them, and because g_1 and g_2 are linear, the problem can be defuzzified to

[3]Otherwise, different α- and β-cuts should be needed and the problem would become more difficult

obtain the following crisp LP problem:

$$\begin{aligned} \max z &= g_1(\tilde{\mathbf{c}})\mathbf{x} \\ s.t.: g_2(\tilde{\mathbf{a}}_i)\mathbf{x} &\leq g_2(\tilde{\mathbf{b}}_i) + g_2(\tilde{\mathbf{t}}_i)(1-\alpha), \quad i = 1, ..., m \\ \mathbf{x} &\geq \mathbf{0} \end{aligned}$$

The function implementing this approach is called `GFLP`. It receives the two linear ordering functions to be used (one for the objective function and the other for the constraints). They must be one of the functions described in section 3.2. Since the problem has to be solved for a fixed α and then the Representation Theorem is used, the function samples $\alpha \in [0, 1]$ with a user-specified step size.

The function will be demonstrated with the following example:

$$\begin{aligned} \max z &= (1,3,4,5)x_1 + (0,1,1,2)x_2 \\ s.t.: (0,2,2,3.5)x_1 + (0,1,1,4)x_2 &\leq (2,2,2,3) + (1,2,2,3)(1-\alpha) \\ (3,5,5,6)x_1 + (1.5,2,2,3)x_2 &\leq 12 \\ x_1 &\geq 0, x_2 \geq 0 \end{aligned}$$

The R commands below create the necessary objects:

```
> objective<-c(TrapezoidalFuzzyNumber(1,3,4,5),
+              TrapezoidalFuzzyNumber(0,1,1,2))
> A<-matrix(c(TrapezoidalFuzzyNumber(0,2,2,3.5),
+             TrapezoidalFuzzyNumber(3,5,5,6),
+             TrapezoidalFuzzyNumber(0,1,1,4),
+             TrapezoidalFuzzyNumber(1.5,2,2,3)), nrow= 2)
> dir = c("<=", "<=")
> b<-c(TrapezoidalFuzzyNumber(2,2,2,3), 12)
> t<-c(TrapezoidalFuzzyNumber(1,2,2,3),0)
```

The example employs the *average index* (which receives two additional parameters λ and t) for the objective function, and Adamo for the constraints. As the latter only requires one parameter, it can be passed directly without using a tagged vector.

```
> sal<-GFLP(objective, A, dir, b, t, TRUE, "Average",
+           ordf_obj_param=c(lambda=0.5, t=3),
+           ordf_res="Adamo", ordf_res_param = 0.5)
> sal
```

```
     beta x1        x2 objective
[1,] 0    1.818182  0  ?
[2,] 0.25 1.590909  0  ?
[3,] 0.5  1.363636  0  ?
[4,] 0.75 1.136364  0  ?
[5,] 1    0.9090909 0  ?
```

```
> sal[,"objective"]

[[1]]
Trapezoidal fuzzy number with:
   support=[1.81818,9.09091],
      core=[5.45455,7.27273].
# ... output omitted for elements 2 to 5
```

4. Conclusions and further work

An R package for solving FLP problems has been presented and demonstrated in simple use cases. It can deal with fuzzy constraints, fuzzy costs and a fuzzy technology matrix, and provides specific functions for solving each type of problem. The computations are done with TrFNs as they ease the application of several theoretical results. Our code relies on packages `FuzzyNumbers` for creating and working with TrFNs, and `ROI` for solving the crisp LP problems in which the FLP problems are transformed.

To the best of our knowledge, this is the first open-source implementation of FLP solving methods, and possibly the only one available in a modern language, as R is. It has been developed as a library of functions, which broadens its usability. Nevertheless, much more can still be done in this direction, such as incorporating more solving techniques for other types of FLP problems, and spanning the functionality to fuzzy non-linear optimization, such as Fuzzy Quadratic Programming (FQP).

Acknowledgments

David A. Pelta and José Luis Verdegay want to acknowledge Ronald Yager for his support, help and sincere friendship. This work is supported by projects TIN2011-27696-C02-01 from the Spanish Ministry of Science and Innovation, P11-TIC-8001 from the Andalusian Government, and FEDER funds. The first author acknowledges an FPU scholarship from the Spanish Ministry of Education.

References

Adamo JM (1980). "Fuzzy decision trees." *Fuzzy Sets and Systems*, **4**, 207–219.

Alefeld G, Herzberger J (1984). *Introduction to interval computations*. Academic Press, NY.

Bellman R, Zadeh L (1970). "Decision Making in a Fuzzy Environment." *Management Science 17 (B) 4*, pp. 141–164.

Bitran G (1985). "Linear multiple objective Problems with Interval Coefficients." *Management Science*, **26(7)**, 694–706.

Cadenas J, Liern V, Sala R, Verdegay J (2010). *Fuzzy Optimization*, chapter Fuzzy Linear Programming in Practice: An Application to the Spanish Football League, pp. 503–528. Studies in Fuzziness and Soft Computing. Springer.

Cadenas J, Pelta D, Pelta H, Verdegay J (2004). "Application of fuzzy optimization to diet problems in Argentinean farms." *European Journal of Operational Research*, **158**, 218–228.

Cadenas J, Pelta D, Verdegay J (2003). "Introducing SACRA: a decision support system for the construction of cattle diets." In *Applied Decision Support with Soft Computing*, pp. 391–401. Springer.

Cadenas JM, Verdegay JL (1995). "PROBO: an interactive system in fuzzy linear programming." *Fuzzy Sets and Systems*, **76**, 319–332.

Cadenas JM, Verdegay JL (1999). *Optimization models with imprecise data (in Spanish)*. Servicio de Publicaciones, University of Murcia.

Dantzig GB (1987). "Origins of the simplex method." *Technical Report SOL 87-5*, Department of Operations Research, Stanford University, Stanford, CA.

Delgado M, Herrera F, Verdegay JL, Vila MA (1993). "Post-optimality analisys on the membership function of a fuzzy linear programming problem." *Fuzzy Sets and Systems*, **53**, 289–297.

Delgado M, Verdegay JL, Vila MA (1987). "Imprecise costs in mathematical programming problems." *Control and Cybernetics*, **16**(2), 113–121.

Dubois D, Prade H (1978). "Operations on Fuzzy Numbers." *International Journal of Systems Science*, **9**, 613–626.

Dubois D, Prade H (1980). *Fuzzy Sets and Systems. Theory and Applications*. Academic Press.

Dubois D, Prade H (1983). "Ranking fuzzy numbers in the setting of possibility theory." *Information Sciences*, **30**(3), 183 – 224.

Fedrizzi M, Kacprzyk J, Verdegay J (1991). "A Survey of Fuzzy Optimization and Mathematical Programming." In M Fedrizzi, J Kacprzyk, M Roubens (eds.), *Interactive Fuzzy Optimization*, volume 368 of *Lecture Notes in Economics and Mathematical Systems*, pp. 15–28. Springer Berlin Heidelberg.

Gagolewski M (2014). *FuzzyNumbers Package: Tools to Deal with Fuzzy Numbers in R*. http://FuzzyNumbers.rexamine.com/.

González A (1990). "A study of the ranking function approach through mean values." *Fuzzy Sets and Systems*, **35**, 29–41.

Kornbluth JSH, Steuer RE (1981). "Multiple Objective Linear Fractional Programming." *Management Science*, **27**(9), 1024–1039.

Moore RE (1979). *Methods and Applications of Interval Analysis*. SIAM Studies in Applied and Numerical Mathematics, book 2. SIAM.

Negoita CV, Ralescu DA (1975). *Applications of Fuzzy Sets to Systems Analysis*. John Wiley and Sons, Ltd.

Peidro D, Mula J, Poler R (2007). "Supply chain planning under uncertainty: a fuzzy linear programming approach." In *Proceedings of the 2007 IEEE Fuzzy Systems Conference*, pp. 1–6.

Prade H, Yager RR, Dubois D (eds.) (1993). *Readings in Fuzzy Sets for Intelligent Systems*. Morgan Kaufmann Publishers.

R Core Team (2013). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna. ISBN 3-900051-07-0. URL http://www.R-project.org/.

Rommelfanger H (1996). "Fuzzy linear programming and applications." *European Journal of Operational Research*, **92**, 512–527.

Rommelfanger H, Hanuscheck R, Wolf J (1989). "Linear programming with fuzzy objectives." *Fuzzy Sets and Systems*, **29**, 31–48.

Rosenthal RE (2014). *GAMS: A User's Guide*. GAMS Development Corporation, Washington DC.

Sadeghi M, Hosseini HM (2013). "Evaluation of Fuzzy Linear Programming Application in Energy Models." *International Journal of Energy Optimization and Engineering*, **2**(1), 50–59.

Silva RC, Cruz C, Verdegay JL, Yamakami A (2010). "A Survey of Fuzzy Convex Programming Models." In WA Lodwick, J Kacprzyk (eds.), *Fuzzy Optimization*, volume 254 of *Studies in Fuzziness and Soft Computing*, pp. 127–143. Springer Berlin Heidelberg.

Tanaka H, Ichihashi H, Asai F (1984). "A formulation of fuzzy linear programming problems based a comparison of fuzzy numbers." *Control and Cybernetics*, **13**, 185–194.

Tanaka H, Okuda T, Asai K (1974). "On Fuzzy Mathematical Programming." *Journal of Cybernetics*, **3,4**, 37–46.

Theussl S, Meyer D, Hornik K (2010). "Many Solvers, One Interface: ROI, the R Optimization Infrastructure Package." In *useR! conference*, p. 161. URL http://www.r-project.org/conferences/useR-2010/abstracts/_Abstracts.pdf.

Tsakiris G, Spiliotis M (2004). "Fuzzy Linear Programming for problems of water allocation under uncertainty." *European Water*, **7-8**, 25–37.

Vasant PM (2003). "Application of Fuzzy Linear Programming in Production Planning." *Fuzzy Optimization and Decision Making*, **2**(3), 229–241.

Verdegay JL (1982). "Fuzzy Mathematical Programming." *In: Fuzzy Information and Decision Processes*, pp. 231–237. M.M. Gupta and E. Sánchez (eds).

W N Venables and D M Smith and the R Core Team (2014). *An Introduction to R, version 3.1.2*. R Foundation for Statistical Computing, Vienna. URL http://cran.r-project.org/doc/manuals/R-intro.pdf.

Werners B (1987). "An interactive fuzzy programming system." *Fuzzy Sets and Systems*, **23**, 131–147.

Yager RR (1978). "Ranking fuzzy subsets over the unit interval." In *Proc. of the IEEE Conference on Decision and Control*, pp. 1435–1437.

Zadeh L (1975). "The Concept of a Linguistic Variable and its Applications to Approximate Reasoning, part I, II and III." *Information Sciences*, **8:199-249, 8:301-357, 9:43-80**.

Zimmermann HJ (1976). "Description and Optimization of fuzzy Systems." *International Journal of General Systems*, **2**, 209–215.

Zimmermann HJ (1978). "Fuzzy programming and linear programming with several objective functions." *Fuzzy Sets and Systems*, **1**(1), 45–55.

Affiliation:

Pablo J. Villacorta, Carlos A. Rabelo, David A. Pelta, José L. Verdegay
Models of Decision and Optimization (MODO) Research Group
CITIC-UGR, Department of Computer Science and Artificial Intelligence
University of Granada
18071 Granada, Spain
E-mail: pjvi@decsai.ugr.es, carabelo@gmail.com, dpelta@decsai.ugr.es, verdegay@decsai.ugr.es
URL: http://decsai.ugr.es/~pjvi, http://decsai.ugr.es/~dpelta, http://decsai.ugr.es/~verdegay, http://tic169.ugr.es